Pointer Network

Refinements to Beam search

2020/7/25

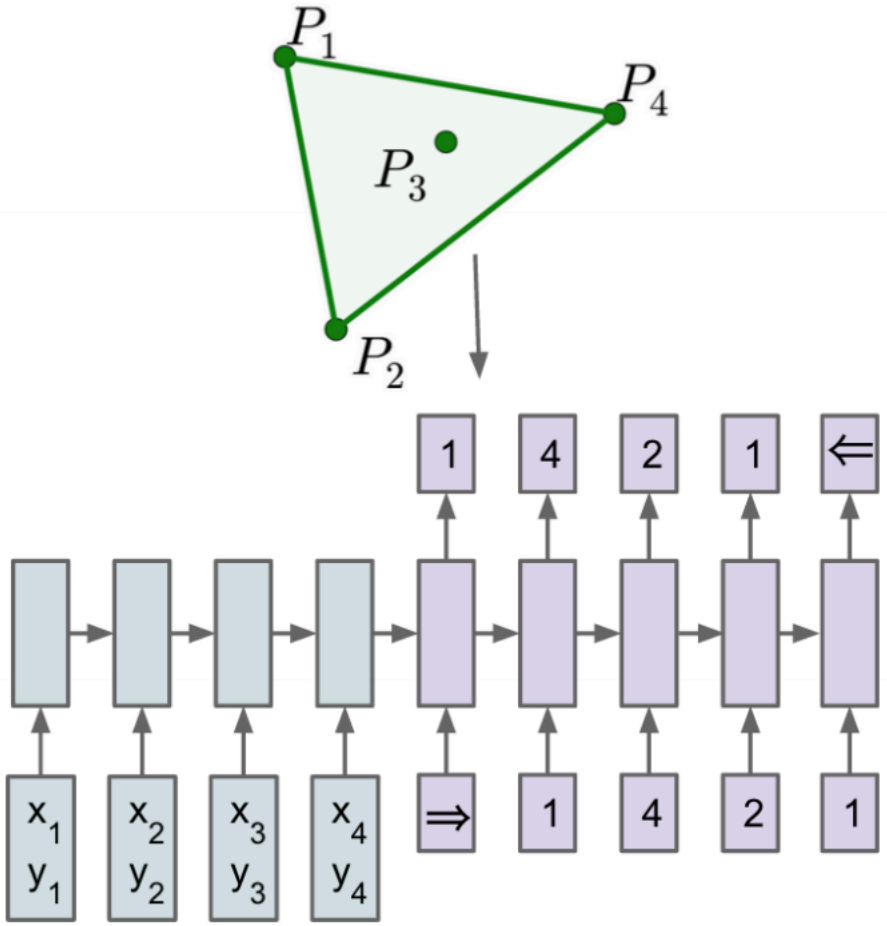
NeuHub
AI Links All

Goals To Take Away:

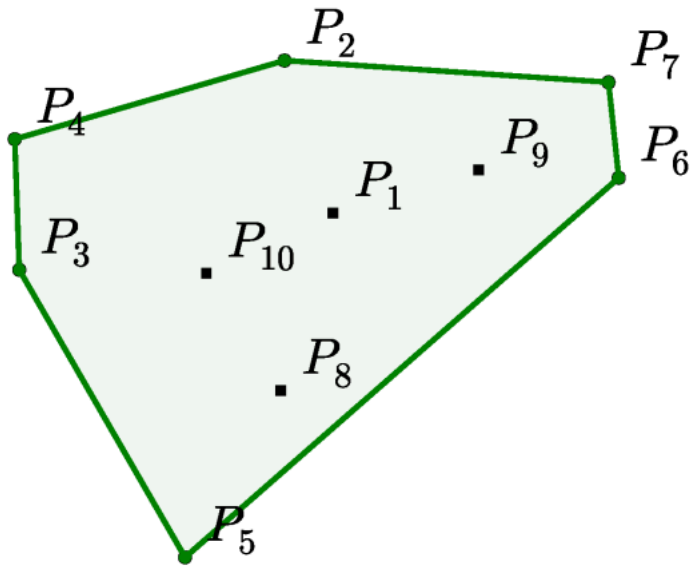
1. Why Pointer Network
2. Structure of Pointer Network
3. How to process language generation by incorporating Pointer Network
4. Get To The Point: Summarization with Pointer-Generator Networks
5. CopyNet
6. Product abstract generation through pointer network
7. Beam search quick Review
8. Beam search problem
9. Length normalization
10. Coverage normalization
11. End of sentence normalization

NeuHub
AI Links All

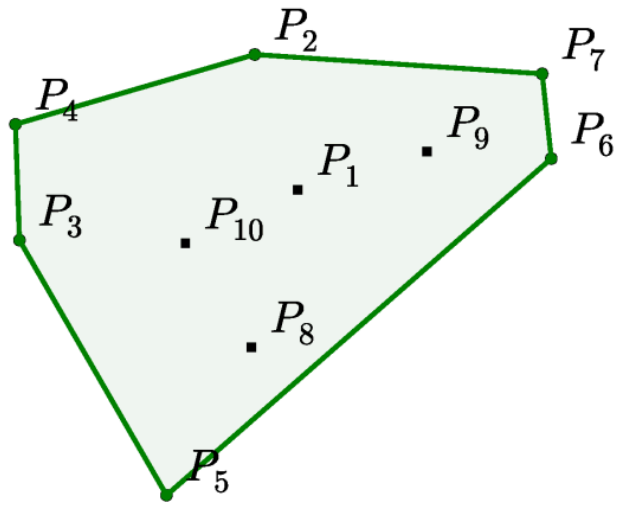
Why Pointer Network



Review : Seq2Seq + Attention



Why Pointer Network

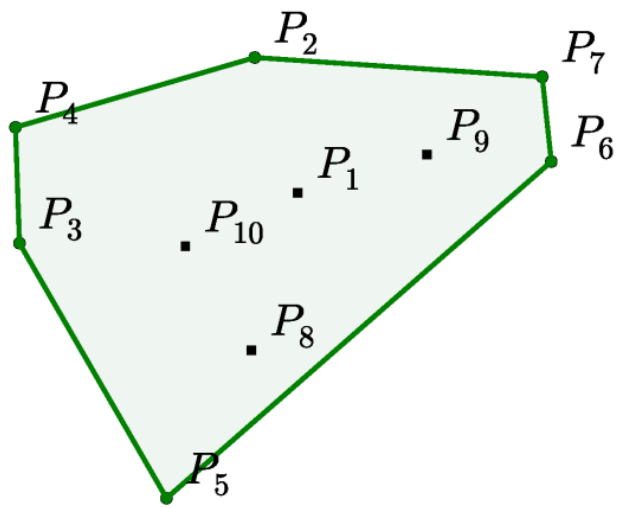


(a) Input $\mathcal{P} = \{P_1, \ldots, P_{10}\}$, and the output sequence $\mathcal{C}^{\mathcal{P}} = \{\Rightarrow, 2, 4, 3, 5, 6, 7, 2, \Leftarrow\}$ representing its convex hull.

- Convex Hull
- Delaunay Triangulation
- Travelling Salesman Problem

NeuHub
AI Links All

Why Pointer Network



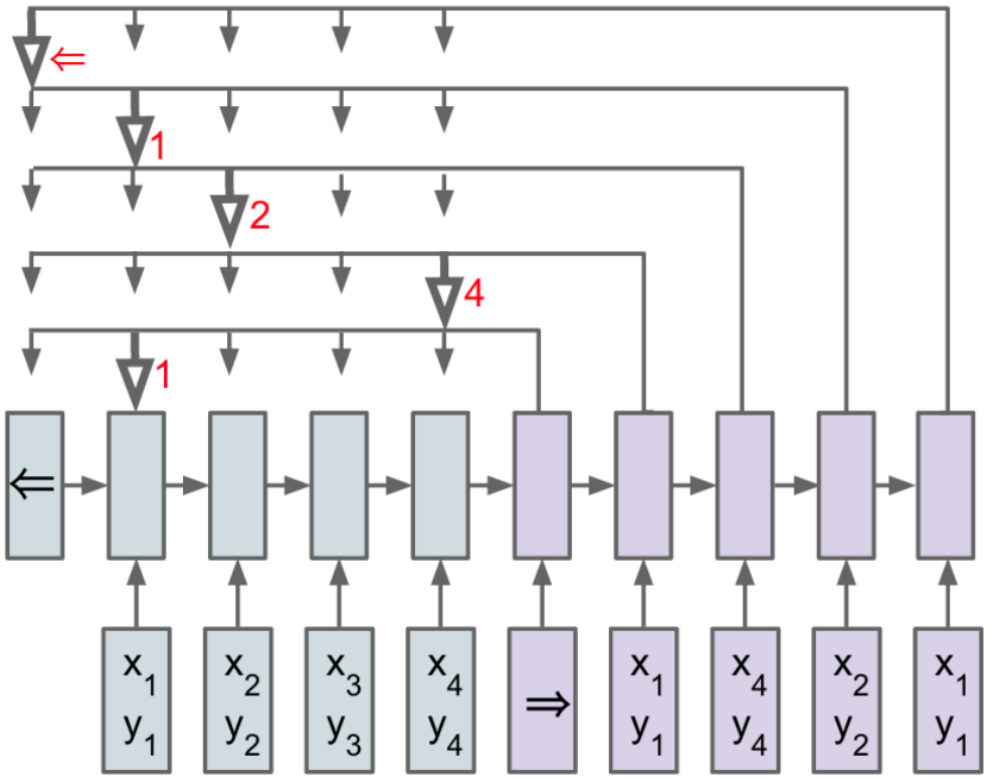
(a) Input $\mathcal{P} = \{P_1, \ldots, P_{10}\}$, and the output sequence $\mathcal{C}^{\mathcal{P}} = \{\Rightarrow, 2, 4, 3, 5, 6, 7, 2, \Leftarrow\}$ representing its convex hull.

传统的seq2seq模型是无法解决**输出序列的词汇表**会**随着输入序列长度的改变而改变的问题的**。

如寻找凸包等。因为对于这类问题，输出往往是输入集合的子集。基于这种特点，作者考虑能不能找到一种结构类似编程语言中的指针，每个指针对应输入序列的一个元素，从而我们可以直接操作输入序列而不需要特意设定输出词汇表。

NeuHub
AI Links All

Structure of Pointer Network



(b) Ptr-Net

传统的Att:

$$
\begin{aligned}
u_j^i &= v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in (1, \ldots, n) \\
a_j^i &= \text{softmax}(u_j^i) \quad j \in (1, \ldots, n) \\
d_i' &= \sum_{j=1}^{n} a_j^i e_j
\end{aligned}
$$

其中ej是encoder的隐状态，而di是decoder的隐状态，v,W1,W2都是可学习的参数

Ptr_Net

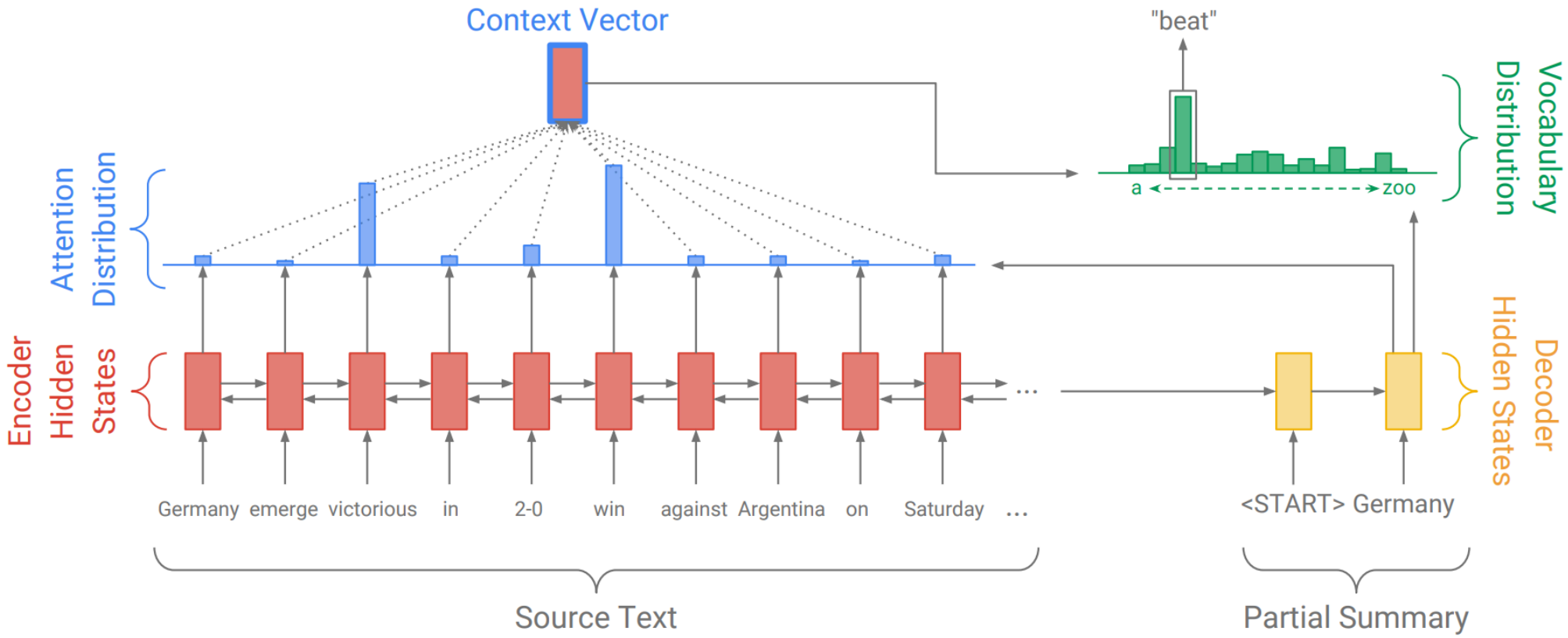
$$
\begin{aligned}
u_j^i &= v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in (1, \ldots, n) \\
p(C_i | C_1, \ldots, C_{i-1}, \mathcal{P}) &= \text{softmax}(u^i)
\end{aligned}
$$

How to process language generation by incorporating Pointer Network ?

1. Language Model ： 自由，灵活，不可控
2. Pointer Net ： 相对可控，信息来源于输入的信息范围
3. 优势：天生的 **复制粘贴利器**
4. Idea: **generate combining language model with Pointer Networks**

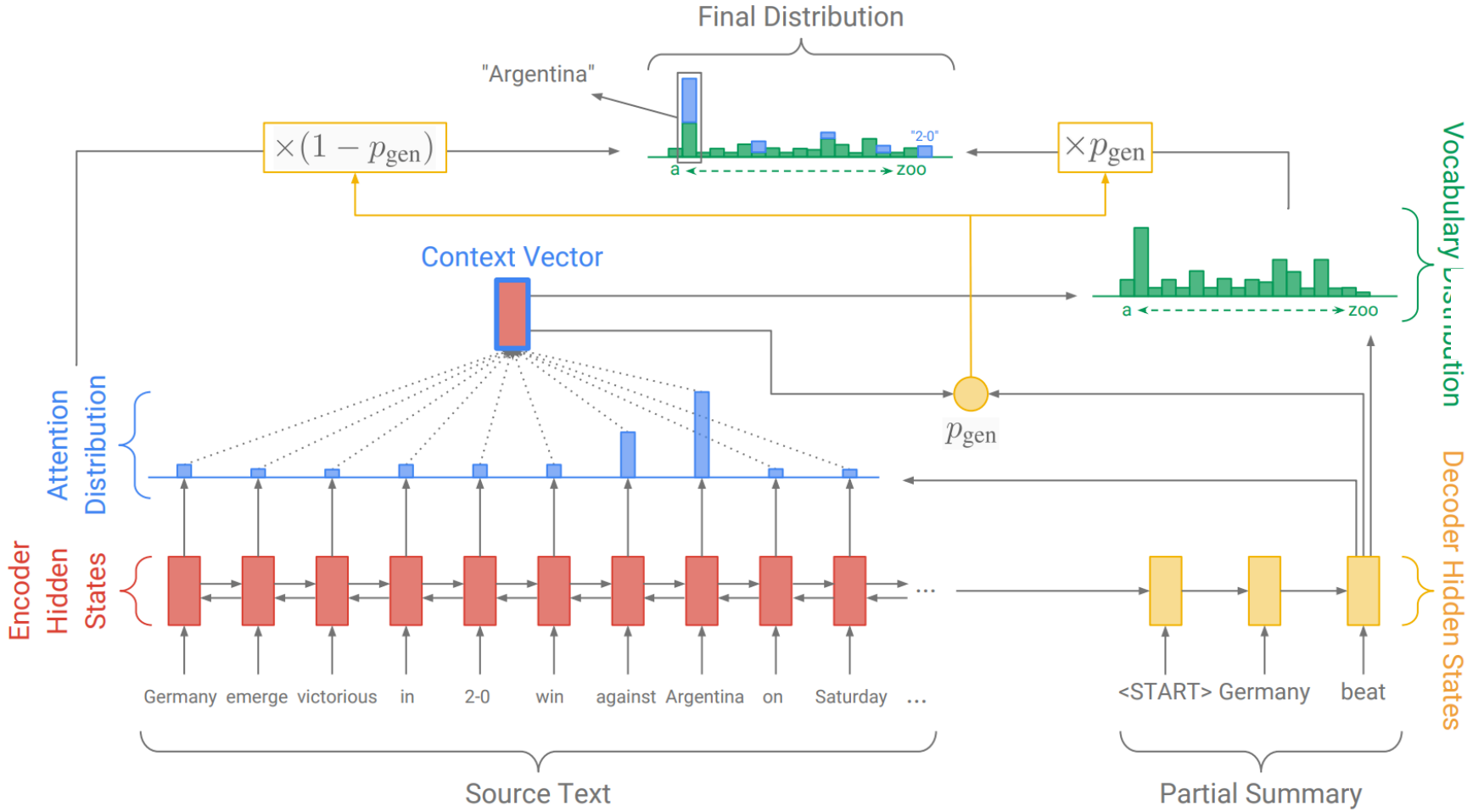
NeuHub
AI Links All

Get To The Point: Summarization with Pointer-Generator Networks



Baseline_Seq2seq + attention

Get To The Point: Summarization with Pointer-Generator Networks



$$e_i^t = v^T \tanh(W_h h_i + W_s s_t + b_{\text{attn}}) \quad (1)$$

$$a^t = \text{softmax}(e^t) \quad (2)$$

$$h_t^* = \sum_i a_i^t h_i \quad (3)$$

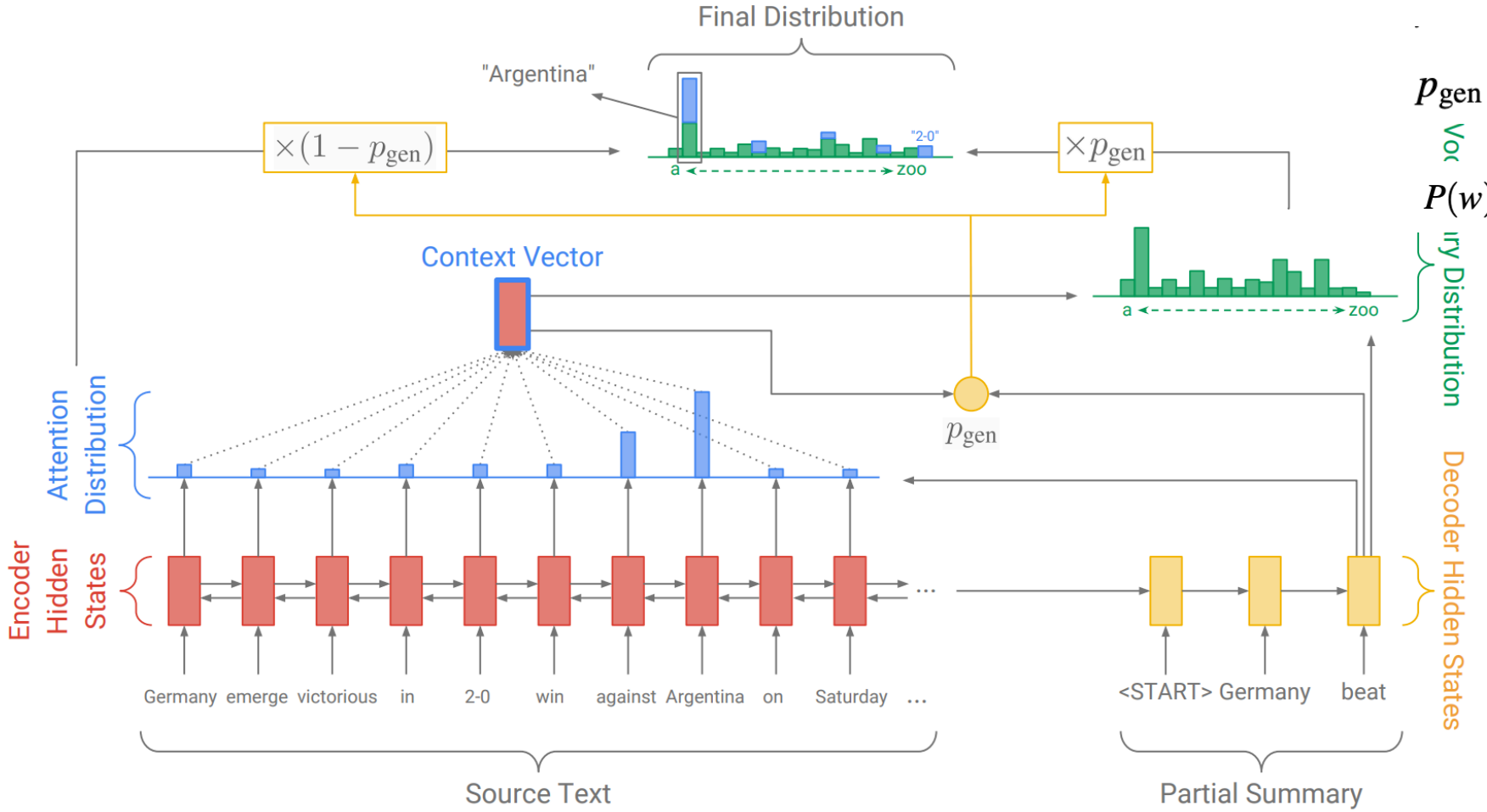
$$P_{\text{vocab}} = \text{softmax}(V'(V[s_t, h_t^*] + b) + b') \quad (5)$$

$$P(w) = P_{\text{vocab}}(w) \quad (5)$$

$$\text{loss}_t = -\log P(w_t^*) \quad (6)$$

AI Links All

Get To The Point: Summarization with Pointer-Generator Networks



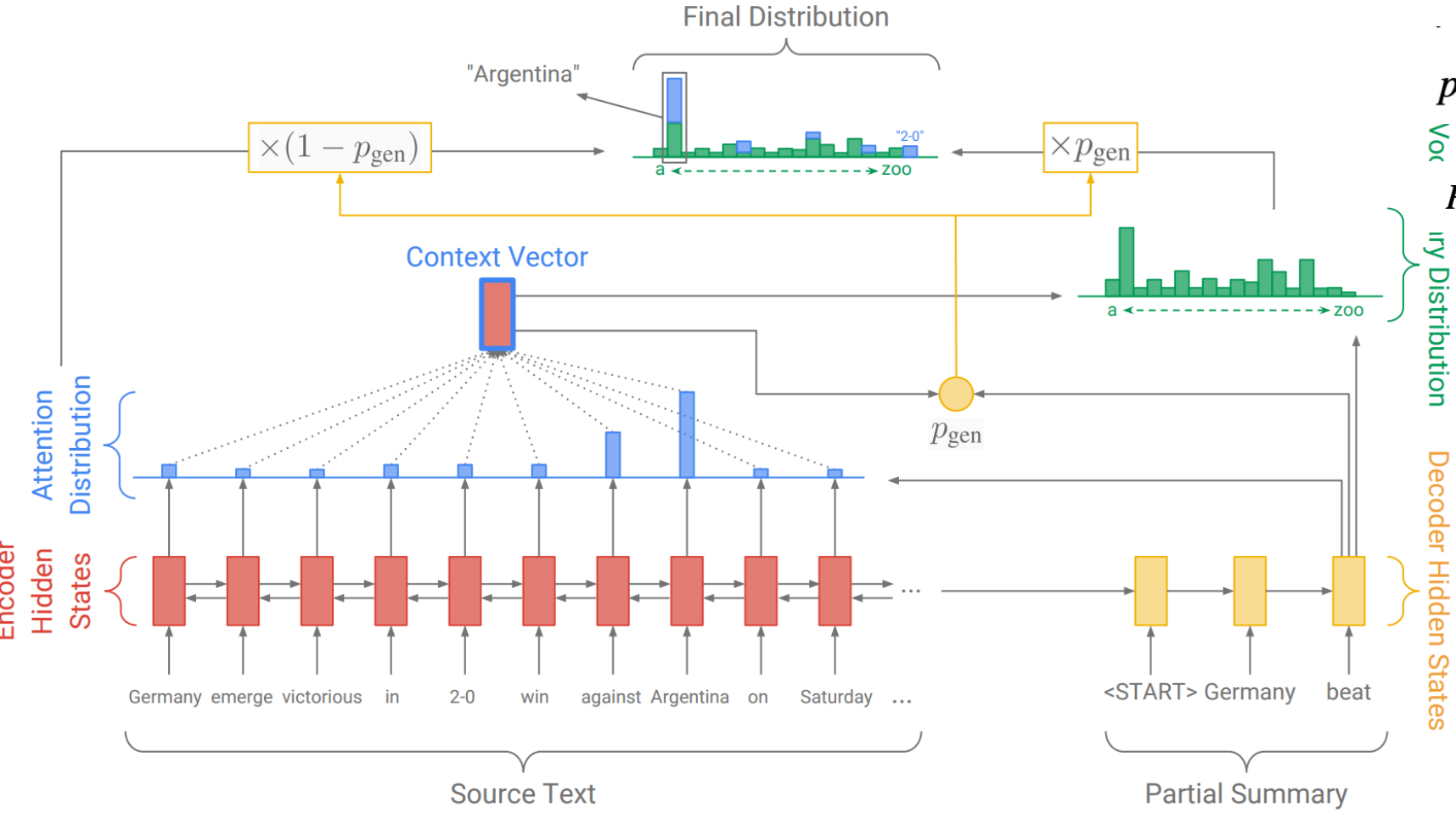
$$p_{\text{gen}} = \sigma(w_{h^*}^T h_t^* + w_s^T s_t + w_x^T x_t + b_{\text{ptr}}) \quad (8)$$

$$P(w) = p_{\text{gen}} P_{\text{vocab}}(w) + (1 - p_{\text{gen}}) \sum_{i:w_i=w} a_i^t \quad (9)$$

What if OOV ?

veuHub AI Links All

Get To The Point: Summarization with Pointer-Generator Networks



$$p_{\text{gen}} = \sigma(w_{h^*}^T h_t^* + w_s^T s_t + w_x^T x_t + b_{\text{ptr}}) \quad (8)$$

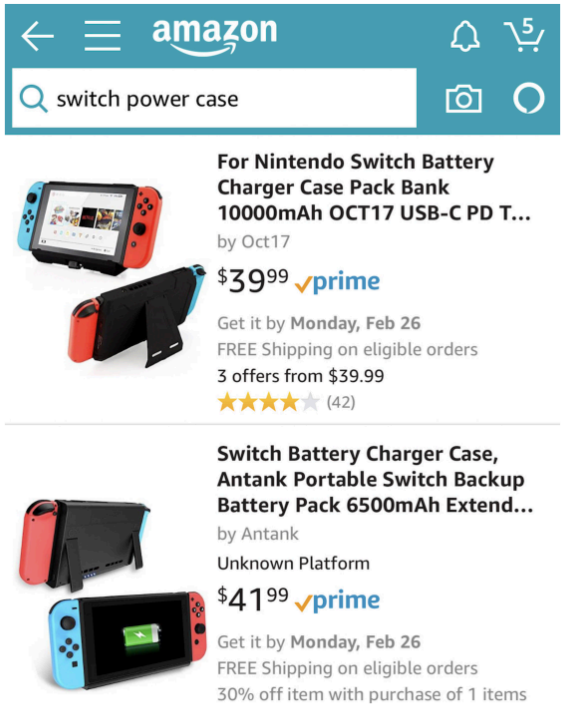
$$P(w) = p_{\text{gen}} P_{\text{vocab}}(w) + (1 - p_{\text{gen}}) \sum_{i:w_i=w} a_i^t \quad (9)$$

What if OOV ?

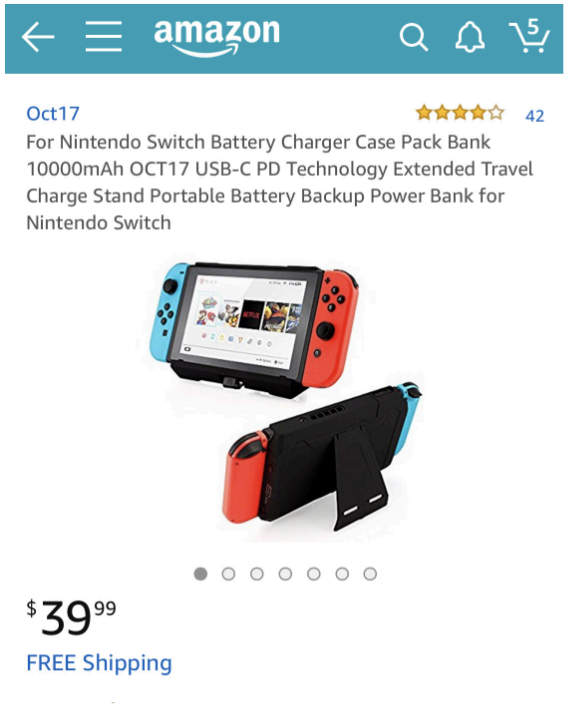
Considering a word w may appear multiple times in the input sequence $\mathcal{S}$, we define the *output distribution* of word w by summing probability mass from all corresponding parts of the attention distribution, as in [38]:

$$p(y_t = w|\mathcal{S}, y_{<t}) = \sum_{i:w_i=w} a_{ti}$$

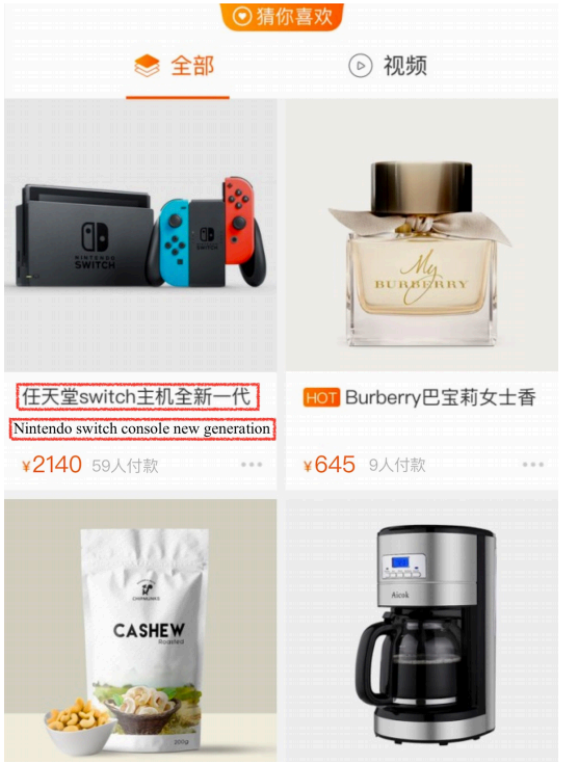
Multi-Source Pointer Network for Product Title Summarization


amazon
switch power case
For Nintendo Switch Battery Charger Case Pack Bank 10000mAh OCT17 USB-C PD T...
by Oct17
$39.99 prime
Get it by Monday, Feb 26
FREE Shipping on eligible orders
3 offers from $39.99
(42)
Switch Battery Charger Case, Antank Portable Switch Backup Battery Pack 6500mAh Extend...
by Antank
Unknown Platform
$41.99 prime
Get it by Monday, Feb 26
FREE Shipping on eligible orders
30% off item with purchase of 1 items

(a) Search page on Amazon


amazon
Oct17
42
For Nintendo Switch Battery Charger Case Pack Bank 10000mAh OCT17 USB-C PD Technology Extended Travel Charge Stand Portable Battery Backup Power Bank for Nintendo Switch
$39.99
FREE Shipping

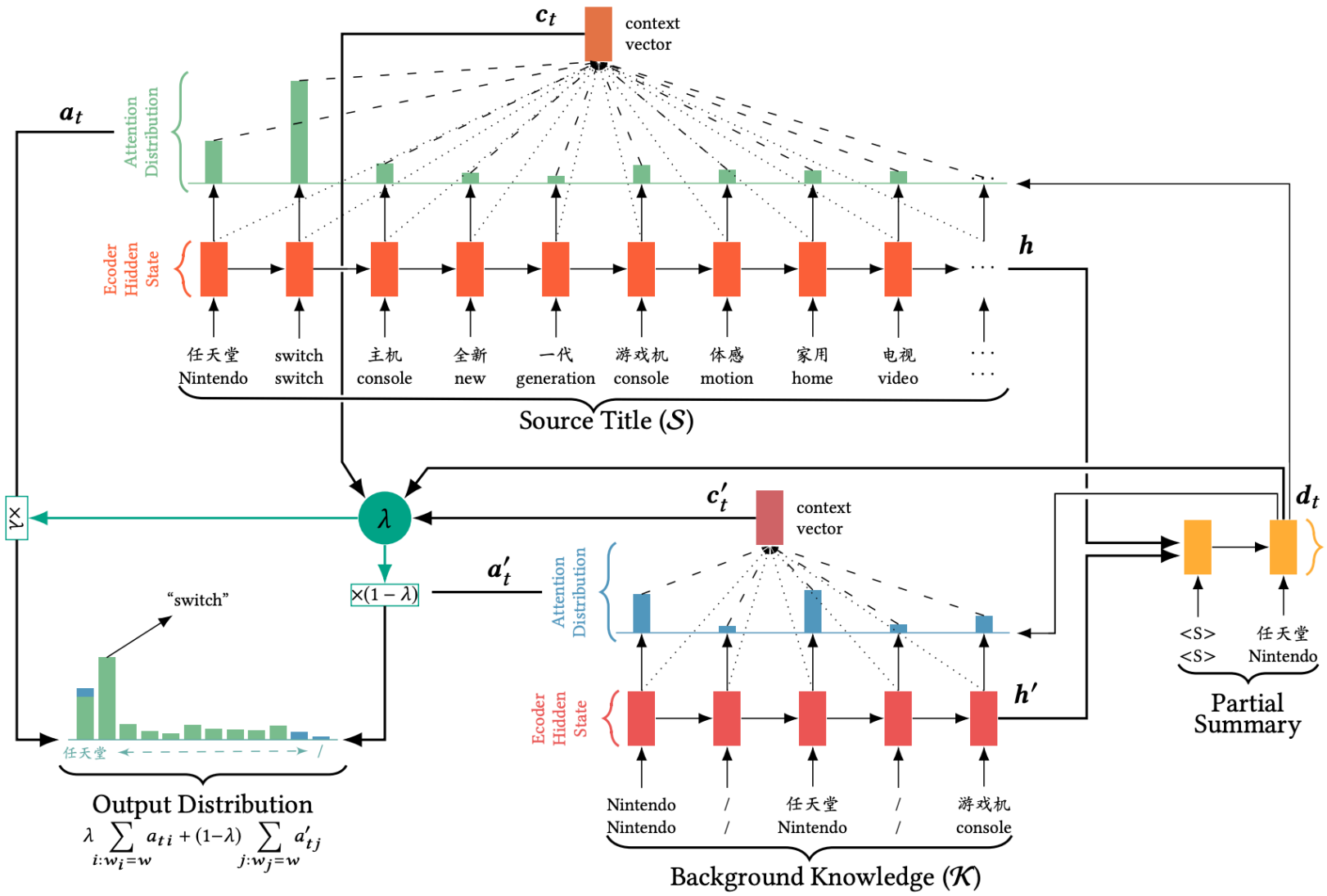
(b) Detail page on Amazon



(c) Recommendation page on Taobao

(d) Detail page on Taobao

Multi-Source Pointer Network for Product Title Summarization



$(h'_1, h'_2, \ldots, h'_M)$ knowledge

$(h_1, h_2, \ldots, h_N)$ source

1. LSTM encode the knowledge and title
2. initialize $d_0 = \mathrm{ReLU}(W_f \cdot [h_N, h'_M])$

3. Attention

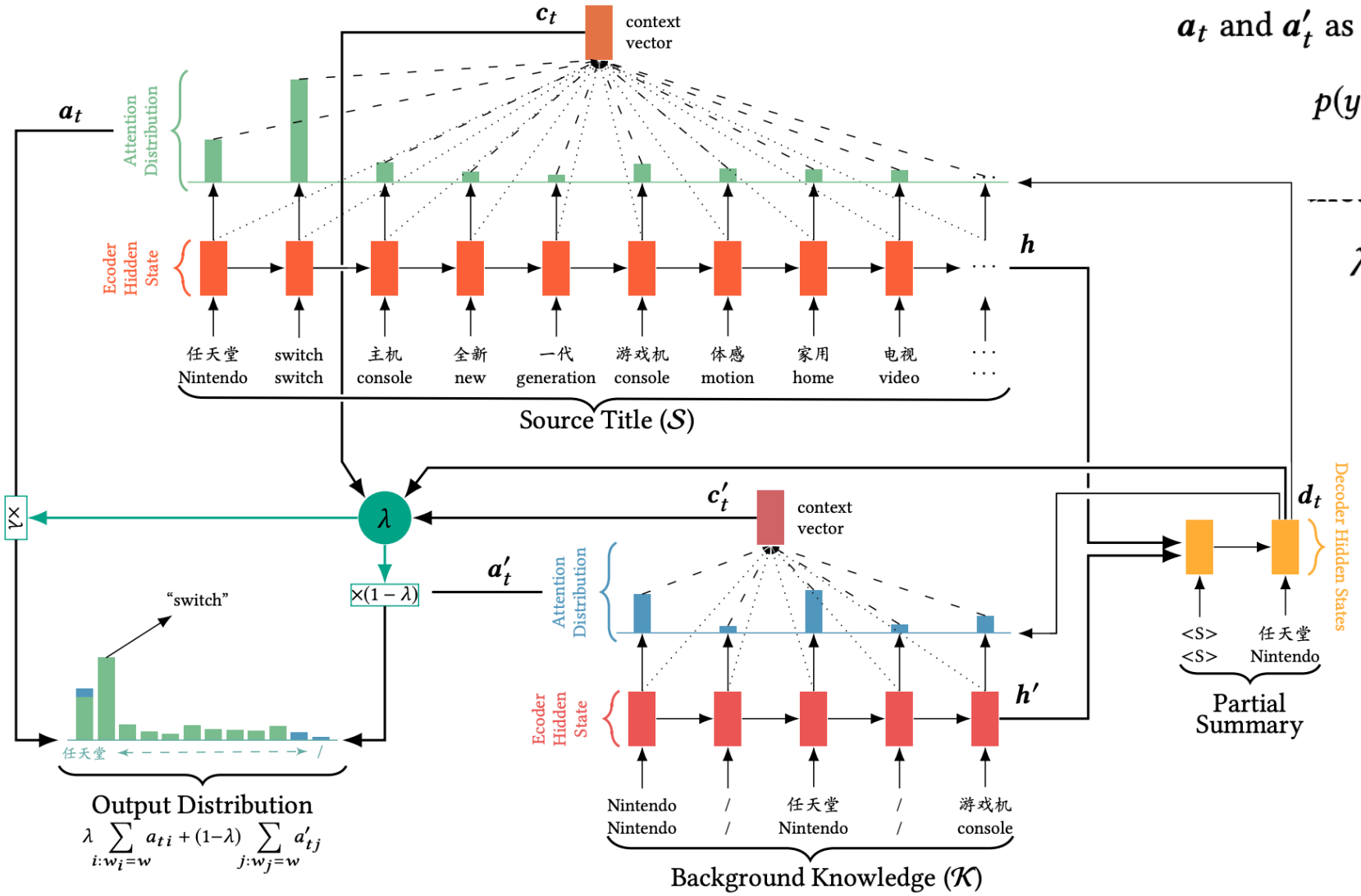
$$u_{ti} = v^\top \tanh(W_h h_i + W_d d_t + b_{\text{attn}})$$
$$u'_{tj} = v'^\top \tanh(W'_h h'_j + W'_d d_t + b'_{\text{attn}})$$
$$a_t = \mathrm{softmax}(u_t), \quad a'_t = \mathrm{softmax}(u'_t)$$

4.decoder output

$$d_t = f(d_{t-1}, y_{t-1}, c_{t-1}, c'_{t-1})$$

$$c_t = \sum_i a_{ti} h_i, \qquad c'_t = \sum_i a'_{ti} h'_i$$

Multi-Source Pointer Network for Product Title Summarization



a_t and a'_t as the final output distribution:

$$p(y_t=w|\mathcal{S},\mathcal{K},y_{<t})=\lambda \sum_{i:w_i=w} a_{ti} + (1-\lambda) \sum_{j:w_j=w} a'_{tj}$$

$$\lambda = \sigma\left(\boldsymbol{w}_d^\top \boldsymbol{d}_t + \boldsymbol{w}_y^\top \boldsymbol{y}_{t-1} + \boldsymbol{w}_c^\top \boldsymbol{c}_t + \boldsymbol{w}_{c'}^\top \boldsymbol{c}'_t\right)$$

$$\mathcal{L} = \frac{1}{T}\sum_{t=0}^{T} -\log p(y_t=w_t^*|\mathcal{S},\mathcal{K},y_{<t})$$

NeuHub AI Links All

CopyNet

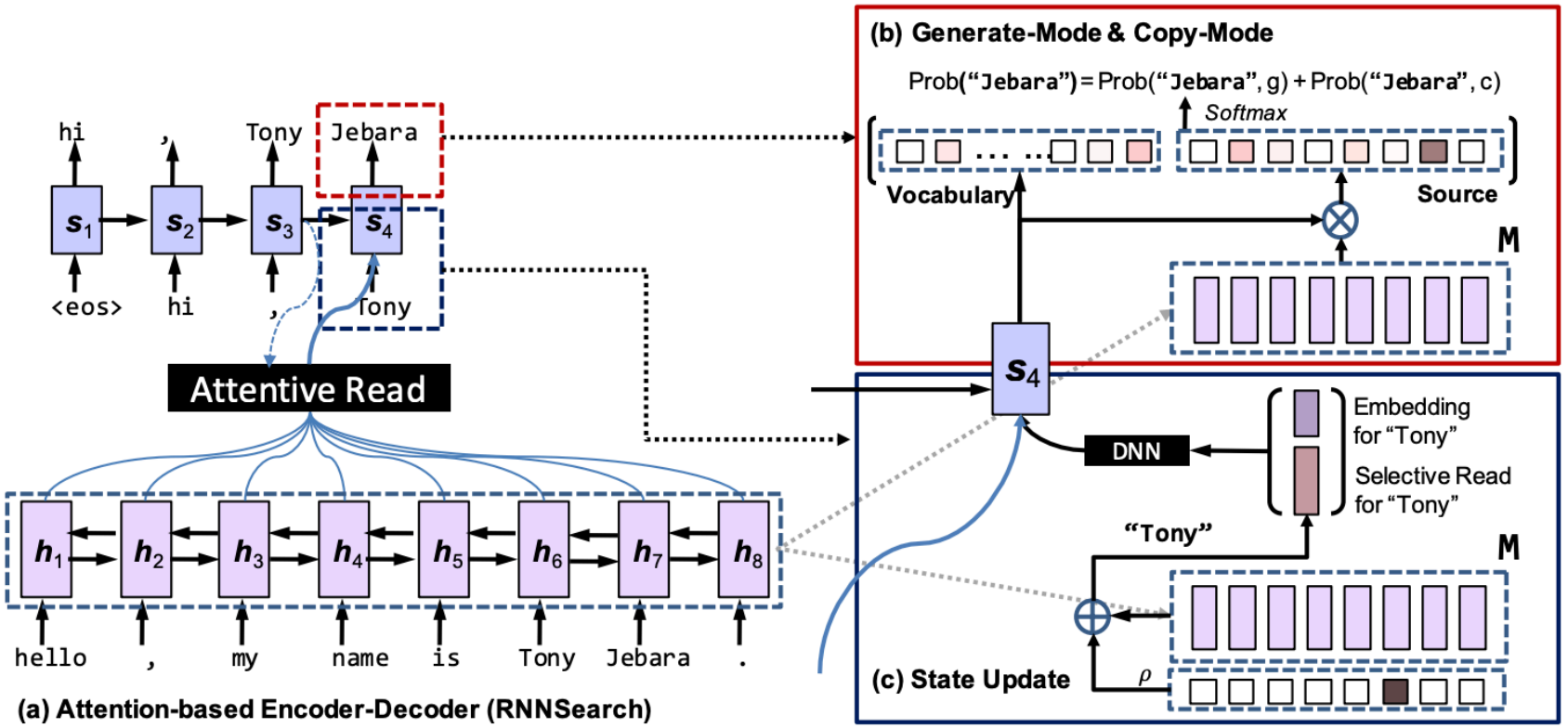
I:	Hello Jack, my name is Chandralekha.
R:	Nice to meet you, Chandralekha.
I:	This new guy doesn't perform exactly as we expected.
R:	What do you mean by "doesn't perform exactly as we expected"?

- Name
- Place
- Organization

NeuHub
AI Links All

CopyNet

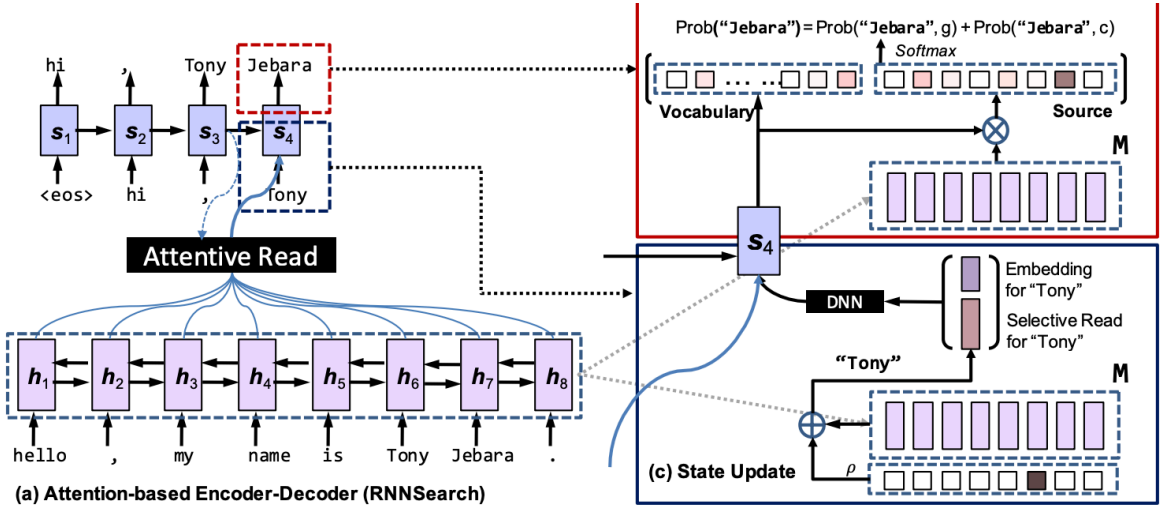
$$p(y_t|\mathbf{s}_t, y_{t-1}, \mathbf{c}_t, \mathbf{M}) = p(y_t, \mathrm{g}|\mathbf{s}_t, y_{t-1}, \mathbf{c}_t, \mathbf{M}) + p(y_t, \mathrm{c}|\mathbf{s}_t, y_{t-1}, \mathbf{c}_t, \mathbf{M}) \quad (4)$$

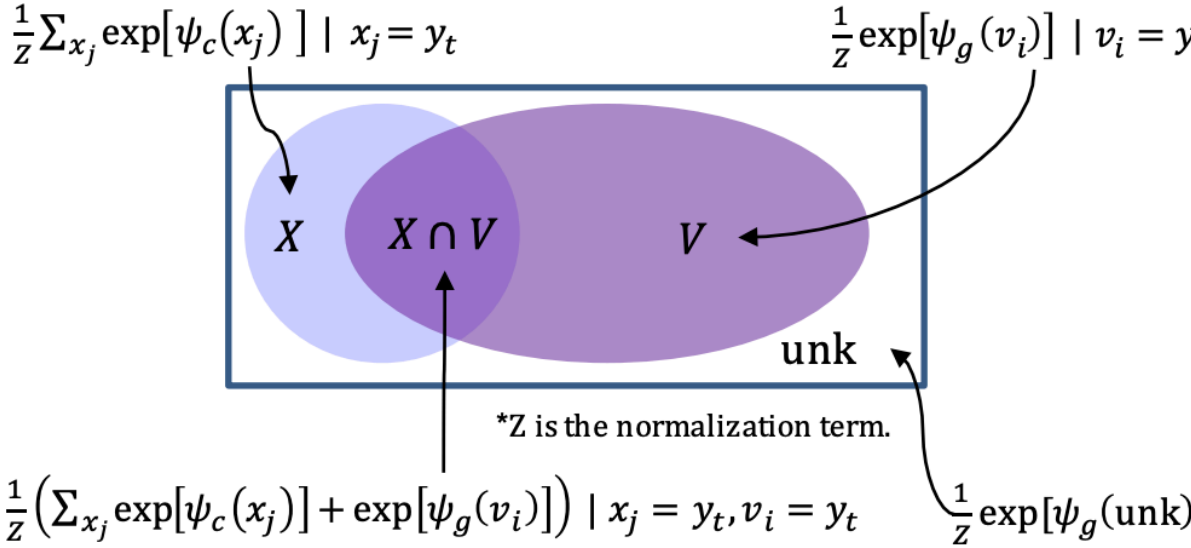


AI Links All

CopyNet

$$p(y_t|\mathbf{s}_t, y_{t-1}, \mathbf{c}_t, \mathbf{M}) = p(y_t, \mathrm{g}|\mathbf{s}_t, y_{t-1}, \mathbf{c}_t, \mathbf{M}) + p(y_t, \mathrm{c}|\mathbf{s}_t, y_{t-1}, \mathbf{c}_t, \mathbf{M}) \quad (4)$$





图中X部分是输入序列的词汇集合，而V部分代表输出词汇集合，当然，通常情况下两个集合都存在交集，而在X和V并集之外的部分就是未知词汇：UNK\

当某个词是输入序列独有的则该词的生成概率为0，复制概率不变；

若某个词是输出词汇表独有的则该词的复制概率为0，而生成概率不变；

若某个词既存在于输入序列又存在于输出词汇表则生成概率和复制概率都不变。

最后，将生成概率和复制概率加和得到最终的概率。

$$p(y_t, \mathrm{g}|\cdot) = \begin{cases} \frac{1}{Z}e^{\psi_g(y_t)}, & y_t \in \mathcal{V} \\ 0, & y_t \in \mathcal{X} \cap \bar{\mathcal{V}} \\ \frac{1}{Z}e^{\psi_g(\mathrm{UNK})} & y_t \notin \mathcal{V} \cup \mathcal{X} \end{cases} \quad (5)$$

$$p(y_t, \mathrm{c}|\cdot) = \begin{cases} \frac{1}{Z}\sum_{j:x_j=y_t} e^{\psi_c(x_j)}, & y_t \in \mathcal{X} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

CopyNet

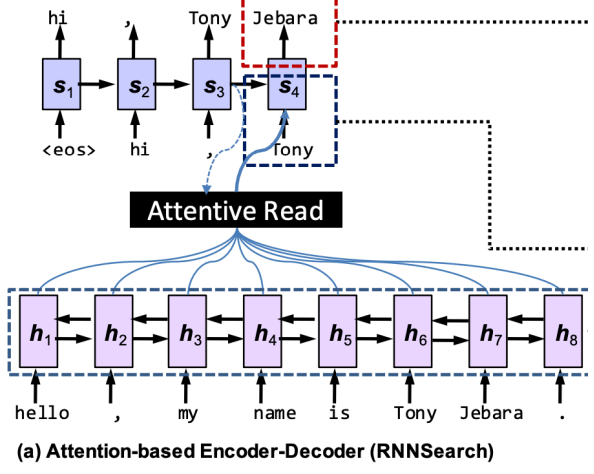
Generate-Mode: The same scoring function as in the generic RNN encoder-decoder (Bahdanau et al., 2014) is used, i.e.

$$\psi_g(y_t = v_i) = \mathbf{v}_i^\top \mathbf{W}_o \mathbf{s}_t, \quad v_i \in \mathcal{V} \cup \text{UNK} \quad (7)$$

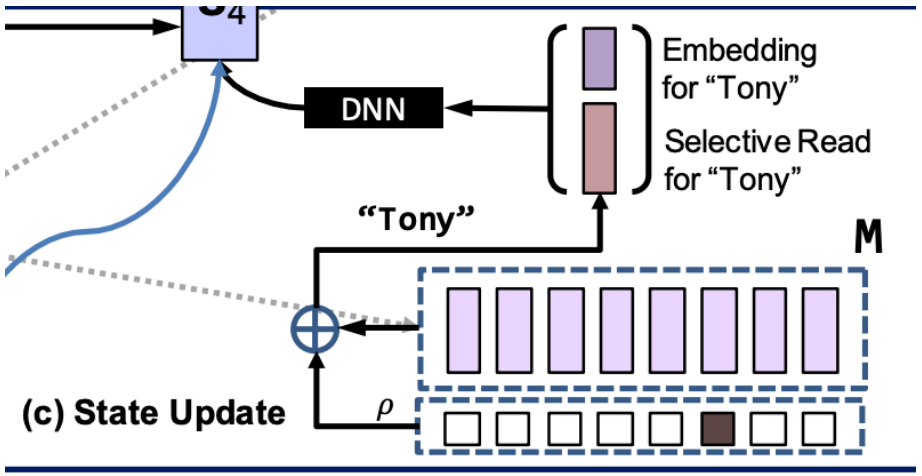
where $\mathbf{W}_o \in \mathbb{R}^{(N+1)\times d_s}$ and $\mathbf{v}_i$ is the one-hot indicator vector for v_i.

Copy-Mode: The score for "copying" the word x_j is calculated as

$$\psi_c(y_t = x_j) = \sigma\left(\mathbf{h}_j^\top \mathbf{W}_c\right)\mathbf{s}_t, \quad x_j \in \mathcal{X} \quad (8)$$



(a) Attention-based Encoder-Decoder (RNNSearch)



(c) State Update

$$\zeta(y_{t-1}) = \sum_{\tau=1}^{T_S} \rho_{t\tau}\mathbf{h}_\tau$$

$$\rho_{t\tau} = \begin{cases} \frac{1}{K}p(x_\tau, \mathbf{c}|\mathbf{s}_{t-1}, \mathbf{M}), & x_\tau = y_{t-1} \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

比如t=4的时候，上一个时刻输出的是 Tony，Tony 这个词对应 source 中t=6的输入，那么就不为0。
通过这种方式，便可以把输入中 Tony 的信息作用到 decoder 中，那么此时时刻拿到的就不仅仅是 Tony 的 emb 信息，还有输入端 Tony 的位置信息以及上下文信息。

NeuHub AI Links All

Beam Search Review

贪心科技 GREEDY TECHNOLOGY

NeuHub
AI Links All

Beam Search Problem

Length normalization

$$\arg\max_y \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

log

$$\arg\max_y \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

$T_y = 1, 2, 3, \dots, 30.$

$$\frac{1}{T_y^{\alpha}} \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

$\alpha = 0.7$ $\alpha = 1$ $\alpha = 0$

AI Links All

Beam Search Problem

Coverage Normalization

2016 年， 华为诺亚方舟实验室的论文提到，机器翻译的时候会存在over translation or under-translation due to attention coverage。 作者提出了coverage-based atttention机制来解决 coverage 问题。 Google machine system 利用了如下的方式进行了length normalization 和 coverage penalty.

$$s(Y,X) = \log(P(Y|X))/lp(Y) + cp(X;Y)$$

$$lp(Y) = \frac{(5+|Y|)^{\alpha}}{(5+1)^{\alpha}}$$

$$cp(X;Y) = \beta * \sum_{i=1}^{|X|} \log(\min(\sum_{j=1}^{|Y|} p_{i,j}, 1.0)),$$

NeuHub
AI Links All

Beam Search Problem

End of sentence Normalization

The score of the end of sentence token is penalized by the following formula:

$$ep(X, Y) = \gamma \frac{|X|}{|Y|}$$

where $|X|$ is the source length, $|Y|$ is the current target length and γ is the end of sentence normalization coefficient `-eos_norm`.

NeuHub
AI Links All